Dual-Level Parallelization of Structural Acoustics Computations

H Allik, [a] **R N Dees,** [a] **T C Oppe,** [b] **D Duffy** [c]

[a] *BBN Technologies, Mystic, CT 06355-3641*
[b] *DMC, ERDC MSRC , Vicksburg, MS 39180-6199*
[c] *CSC, ERDC MSRC, Vicksburg, MS 39180-6199*

Key words: Parallel computing; Structural acoustics; MPI; OpenMP

Abstract

This paper describes the parallelization of a finite element/infinite element structural acoustics code on two high-performance computers. On the SGI Origin 2000 and the IBM Power3 SMP, the code exploits dual-level parallelism, with the Message-Passing Interface (MPI) being used to parallelize coarse-grain computations and OpenMP being used to parallelize several fine-grain computations within each MPI process.

1. Introduction

The solution of large, three-dimensional, submerged structures subjected to time-harmonic loadings is usually accomplished today by solving sets of complex equations resulting from modeling the structure with finite elements and the fluid with finite and infinite acoustic elements. This method leads to a banded set of equations that can be solved much faster than those resulting from the use of the more traditional Boundary Element model of the fluid. However, as one adds to structural complexity, or attempts solutions at higher frequencies, the number of degrees-of-freedom and the bandwidths grow to such an extent that weeks or even months of computer time are required to obtain a solution. Obviously, times like these are unacceptable for performing design calculations. Because of problems of such magnitude, BBN's SARA [1] software was

ported to the SGI Origin 2000 and the IBM Power3 SMP platforms in hopes of attaining reasonable turn-around times after parallelization. The Origin 2000 at the U.S. Army Engineer Research and Development Center (ERDC) Major Shared Resource Center (MSRC) has 128 MIPS R10000 195-MHz processors, while the IBM Power3 SMP has 64 nodes, each with eight 222-MHz processors sharing a 4-GB memory space.

2. Equations of Motion

The coupled equations of motion solved in structural acoustics can be written as

$$\begin{aligned} (K - w^2 M)\, u \quad & -L\, p \quad = f_s \\ -L^T u \quad & + c(H - w^2 Q)\, p = f_f \end{aligned} \tag{1}$$

where in the structural equation K is the stiffness, M the mass, u the displacement vector, and f_s the force vector. Analogous matrices H and Q relate the scalar pressure p to the fluid loadings f_f. In the equation, w is the circular frequency, r is the fluid mass density, and c=1/($r\, w^2$). The L matrix provides the coupling between the structure and the fluid. Equations (1) are complex, symmetric, and banded when the structure and fluid portions are interleaved. Typically, these equations are solved for hundreds of frequencies and for numerous right-hand sides. Having the displacement and pressure results, various types of postprocessing are performed, such as using the Helmholtz equation to calculate field pressures from results on the structure-fluid interface.

3. Method of Parallelization

3.1 Coarse-Grain Parallelism with MPI

SARA employs an inherently parallel algorithm in which the structure is excited at different frequencies, and the work for each frequency is independent of all others.

Hence, using the work of each frequency as the parallel step is an obvious one. Only in recent years, however, has there been enough computer memory and disk space to allow simultaneous solutions of large three-dimensional problems. The part of the code that cycles through the user-specified frequencies, called the "frequency loop," was parallelized using Message-Passing Interface [2], with each MPI process handling a different set of frequencies. To achieve load-balancing, a "boss-worker" strategy was employed in which one MPI process, the "boss," assigns a new frequency to the next available "worker" MPI process that has completed its work for a prior frequency. Once the last frequency has been processed, all MPI processes except one are terminated in preparation for the postprocessing phase.

3.2 Fine-Grain Parallelism with OpenMP

During the frequency loop, the majority of time (more than 90%) is spent in solving the equations, and hence the frontal solver was the first area within an MPI process to be parallelized using OpenMP [3] threads. A frontal solver is a specialized form of Gaussian elimination that organizes the elements of the mesh along a series of computational wavefronts selected in such a way that the front "width," or maximum population of nodes in a front, is minimized. This process is analogous to reordering the rows and columns of a matrix to reduce its bandwidth, which in turn reduces the memory and number of operations required by Gaussian elimination. Nodes are activated in the front when a new element is assembled, but as soon as a node is fully summed, it is eliminated. The front then consists of all active nodes, and the inactive ones have either not appeared or have already been eliminated. The global linear system is never completely assembled in memory, but instead the assembly and elimination are

alternated, with the factors written to disk in a scratch file. Hence the frontal method is an out-of-core solver. The factors are read later in the back-substitution phase when the nodes (and fronts) must be accessed in reverse order.

A version of the frontal solver was effectively parallelized for 6-8 processors using OpenMP. Once an equation is ready to be eliminated, every coefficient in the active front and in the right-hand-side vectors needs to be modified. These are totally independent calculations. The matrix and right-hand-side information can be partitioned into nearly equal pieces and the work assigned to different processors. The equation to be eliminated is made available to every processor, and each one performs the elimination operation, but with different starting and ending indices.

Once the major time-consuming operations have been effectively parallelized, some of the lesser operations in the code become significant. Several of these were parallelized at the loop level using OpenMP. These included several "prefront" calculations, used to symbolically factorize the matrix prior to numerical factorization, and some parts of the front-width minimization routine. Finally, OpenMP was used to parallelize the calculation of field pressures using the Helmholtz integral equation in the postprocessing phase.

3.3 Disk Files

During the course of a SARA run, several files are written to disk, and their use has to be coordinated among the MPI processes. Most files are guarded so that only one MPI

process would write to them. Many of these files, however, are internal binary "write/read" files that needed to be subsequently read by all the MPI processes. Thus, some amount of synchronization was necessary to prevent the reading of an incompletely written file. An example is the large disk file that contains the matrices of all the frequency-independent elements in the model. These matrices are calculated and placed in a file during the first frequency. During subsequent frequencies, the file is read and the system matrix built using the new frequency.

The largest file is the internal binary scratch file needed by the frontal solver. Each "worker" MPI process required its own version of this file, since each one was generating and solving its own set of linear systems. The size of these frontal solver scratch files effectively limits the number of MPI processes that can be used in a large run, and thus arose the need to parallelize the computations within each MPI process. This "dual-level" parallelism allowed each MPI process to complete its work more quickly, allowing the processing of more frequencies in a given time without using more MPI processes and thus more disk space.

Some postprocessing of results is done in SARA during the frequency loop in order to avoid storing the entire solution. This was not possible when running in parallel. Instead, the MPI processes wrote the solution for each frequency to a disk file whose filename incorporated the number of the frequency. After all frequency solution files were written, a new loop was implemented to postprocess the files in the correct order. This avoided major revisions of the postprocessing segment of the code.

4. Examples

Table 1 contains timing statistics for runs on the ERDC MSRC SGI Origin 2000 and IBM Power3 SMP computers for a finite element model involving 187,571 degrees of freedom. In this run, 11 frequencies were processed. The model was run first sequentially and then using 12 MPI processes and four OpenMP threads per MPI process. Only the solution results include both MPI and OpenMP parallelism; the rest were one-time operations for each MPI process and hence limited to a maximum speedup of four. Tables 2 and 3 show the speedup of the prefront subroutine and the frontal solver, respectively, using OpenMP with four and eight threads.

The results given in Table 4 demonstrate the effectiveness of assigning more threads to a run at the expense of using fewer MPI processes. The problem contains 292,693 degrees of freedom with a front-width of 642. As can be seen, Run 2 takes less wall-clock time than Run 1 and takes approximately one-half the memory and disk space as well. The memory requirements are less since roughly half the number of program executables are running simultaneously. The same problem when run on a single processor Origin took 76.5 hours.

5. Conclusions

The dual-level parallelization of the structural acoustics calculations in SARA using MPI and OpenMP have been very successful in reducing the elapsed running times of large jobs. In one analysis, computations estimated to take more than 1,600 hours on a single-processor machine were completed in 22 hours on an IBM Power3 SMP platform, for an

almost 75-fold reduction. The solution times attainable through parallel computations make it possible to incorporate the results of large-scale models into the design process, making feasible some analyses that were previously impractical.

Acknowledgement

This work was supported in part by a grant of computer time from the Department of Defense High Performance Computing Modernization Program at the ERDC MSRC, Vicksburg, MS.

References

1. Allik H, Moore S, Dees R. Efficient Structural Acoustic Analysis Using Finite and Infinite Elements, ASME 15th Biennial Conference on Vibration and Noise, Volume 3, Part B, Boston, MA, September 1995; 87-92.

2. Snir M, Otto S, Huss-Lederman S, Walker D, Dongarra J. MPI – The Complete Reference: Vol.1, The MPI Core, 2nd ed., Cambridge: The MIT Press, 1998.

3. OpenMP Architecture Review Board, OpenMP Fortran Application Program Interface, Version 1.1, 1999, Available online at http://www.openmp.org.

	Origin Timings(sec)		SMP Timings(sec)	
Phase	Serial	Parallel	Serial	Parallel
Problem Setup	309	198	240	157
Prefront	3402	859	3262	849
Solution	15284	1627	12899	1253
Postprocessing	118	30	173	47
Total Time	19186	2746	16671	3145

Table 1
SGI Origin 2000 and IBM Power3 SMP Timings

Problem	Total dof	Serial Original	Parallel 4 threads	Speedup	Parallel 8 threads	Speedup
1	76607	519	138	3.8	83	6.2
2	113595	1177	307	3.8	178	6.6
3	187571	3308	1005	3.3	563	5.9
4	245519	5728	1434	4.0	784	7.3
5	362340	12400	3409	3.6	1594	7.8

Table 2
Efficiency of OpenMP Threads on the Prefront Subroutine Using the IBM Power3 SMP

Problem	Front-width	Serial Original	Parallel 4 threads	Speedup	Parallel 8 threads	Speedup
1	515	3864	1646	2.3	1488	2.6
2	974	45168	8677	5.2	5887	7.7
3	1068	2018	372	5.4	247	8.1
4	2278	32266	13111	2.5	8131	4.0

Table 3
Speedups of Frontal Solver (Factorization and Forward Substitution) Using OpenMP Threads on the SGI Origin 2000

	Run 1	Run 2
IBM SMP nodes	7	7
MPI processes/ node	4	2
OpenMP threads/ MPI process	2	4
Total number of frequencies	48	48
Max. freq./MPI process	2	4
Total time (hours)	3.28	2.54

Table 4
Effectiveness of Using More Threads per MPI Process